

# Operating System Concepts

# File Systems

IT556- Operating Systems

By:

Dilum Bandara

Dept. of Computer Science & Engineering

University of Moratuwa



# Outline

---

- Why file systems?
- Files
  - Naming, types, structure, access, attributes, operations
- Directories
  - Single level, two level & hierarchical directory systems
  - Operations
- Path Names
- File systems
  - DOS file system, FAT32, NTFS, ext



# Why file systems?

---

- ❑ The data must survive after the termination of the process using it
- ❑ It must be possible to store very large amount of data
- ❑ Multiple processes must be able to access data concurrently
  
- ❑ Solution is to store those data in units called files on disks & other media



# Files

---

- ❑ The data stored in a file must be persistent
- ❑ Files are managed by the OS
- ❑ How they are
  - structured, used, protected & implemented are major concerns
- ❑ The part of the OS that deals with files is known as the File System



# Files - Naming

---

- ❑ The exact rules depends based on the OS
- ❑ However most of them allow files to be:
  - 1-8 characters
  - Digits & several selected symbols
  - Modern ones supports up to 255 characters
  - Examples
    - ❑ osslides, osslides1, osslides2, osslides-1, urgent!
- ❑ Some file systems are case sensitive
  - DOS, Windows – Case insensitive
  - UNIX, Linux – Case sensitive



# Files – Naming etc.

---

- ❑ Many OSs support two-part file systems
  - Parts are separated by a period (.)
  - Example: **<file name>.<extension>**
  - `test.txt`, `os.pdf`, `MyClass.java`, `prog.c`
- ❑ Extension indicates something about the file
- ❑ Not all OSs are aware of extensions
  - UNIX or Linux does not depend on the extension.
  - But some applications may depend on the extension



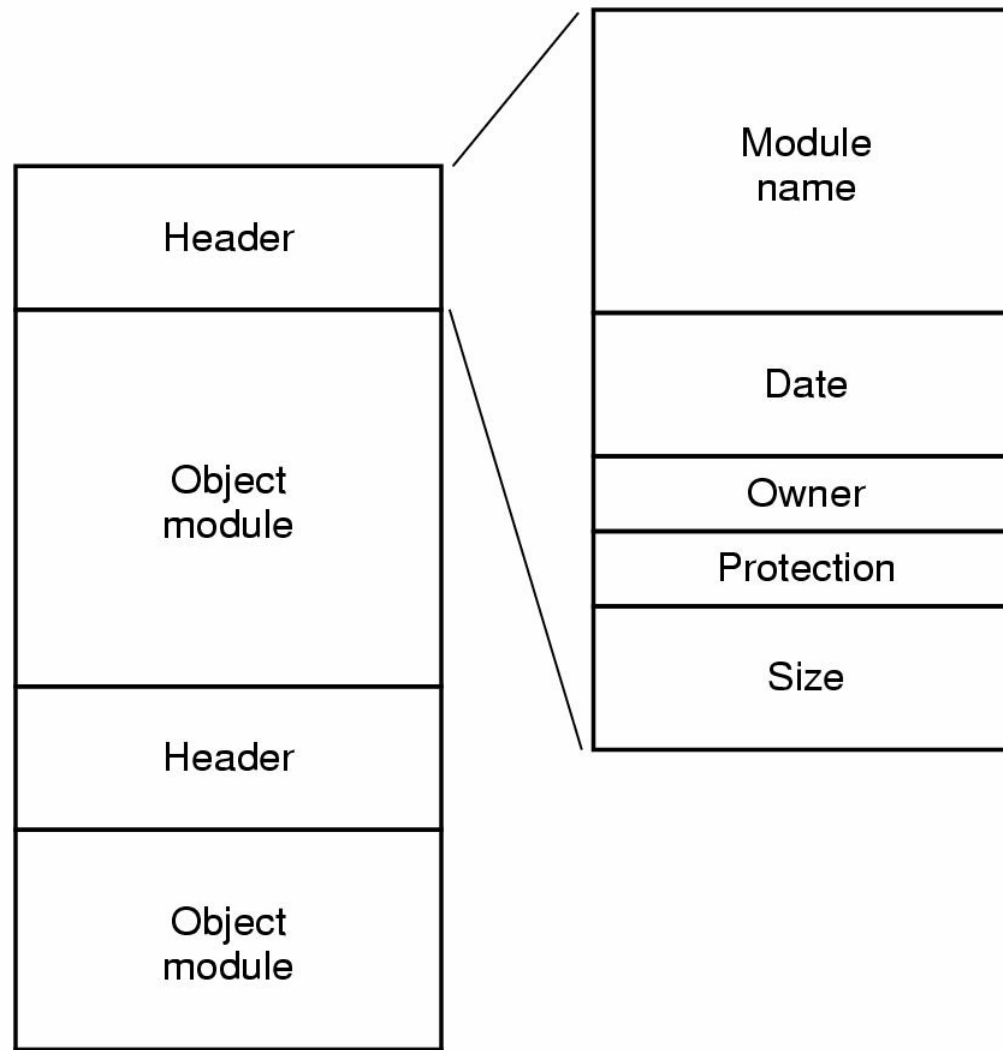
# Files – Types

---

- 2 major types
  - Regular files – ones that contain user data. These are either ASCII or binary
  - Directories – are systems files which are used to maintain the structure of the file system
- In UNIX also has
  - Character files – related to IO & used to model serial IO devices such as terminals & printers
    - /dev/tty, /dev/lp, /dev/net
  - Block files – are used to model disks
    - /dev/hd1, /dev/hd2



# Files - Structure



# File - Access

---

- Can be categorised as:
  1. Sequential access
    - Read all the data starting from the beginning
    - Used in early days with magnetic tapes
    - Example: simple text files
  2. Random access
    - Can read data in a file out of order
    - Were possible with the introduction of magnetic disks
    - Examples: Data bases, movies



# File - Attributes

---

- ❑ A files includes set of other characteristics than just name & an extension
- ❑ some common attributes
  - **Owner** – current owner of file
  - **Creator** – ID of the person who created the file
  - **Protection** – who can access & who can't access
  - **Read only flag** – can it be modified or not
  - **Hidden flag** – display or not when listed
  - **Archive flag** – to be backed up or not
  - **Last modified date, Created date, etc.**



# File - Operations

---

- Different systems allow operations to store & retrieve data from files
  - **Create** – create a new file with no data & set initial attributes
  - **Delete** – remove the file from system freeing up disk space
  - **Open** – before using a files must be open
  - **Read** – after opening a file data can be read
  - **Write** – after opening a file data can be written
  - **Append** - after opening a file data can be written to the end of the file
  - **Close** – when all the access is finished file must be closed



# File – Operations cont...

---

- **Seek** – used in random access a file
- **Get attributes** – get the attributes of a file
- **Set attributes** – set the attributes of a file
- **Rename** – change the name or the extension of a file



# Directories

---

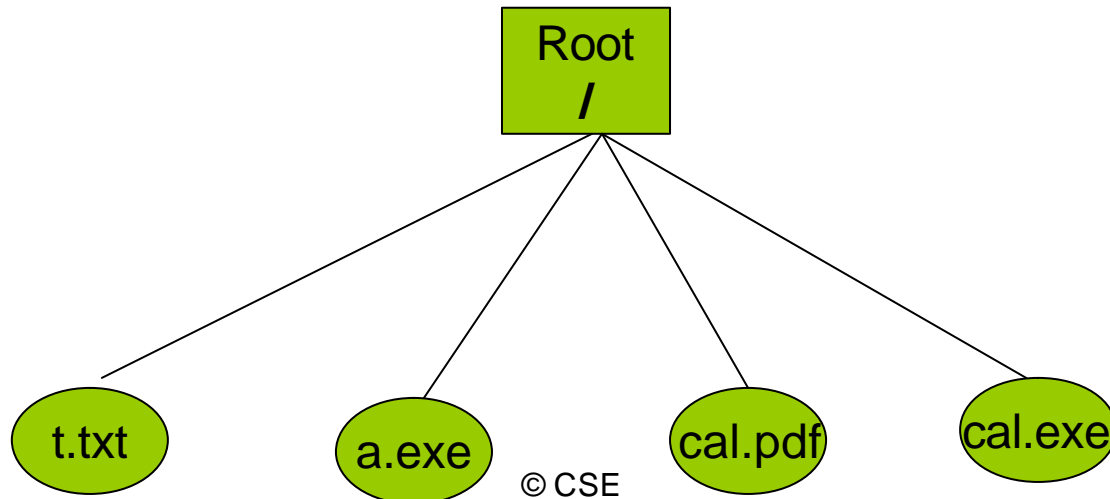
- ❑ Used to organise or keep track of files
- ❑ Are also called folder
- ❑ Most OSs consider even directories as files
  - DOS, UNIX, Linux call directories
  - While MS-Windows call them as Folders



# Directories - Single Level Systems

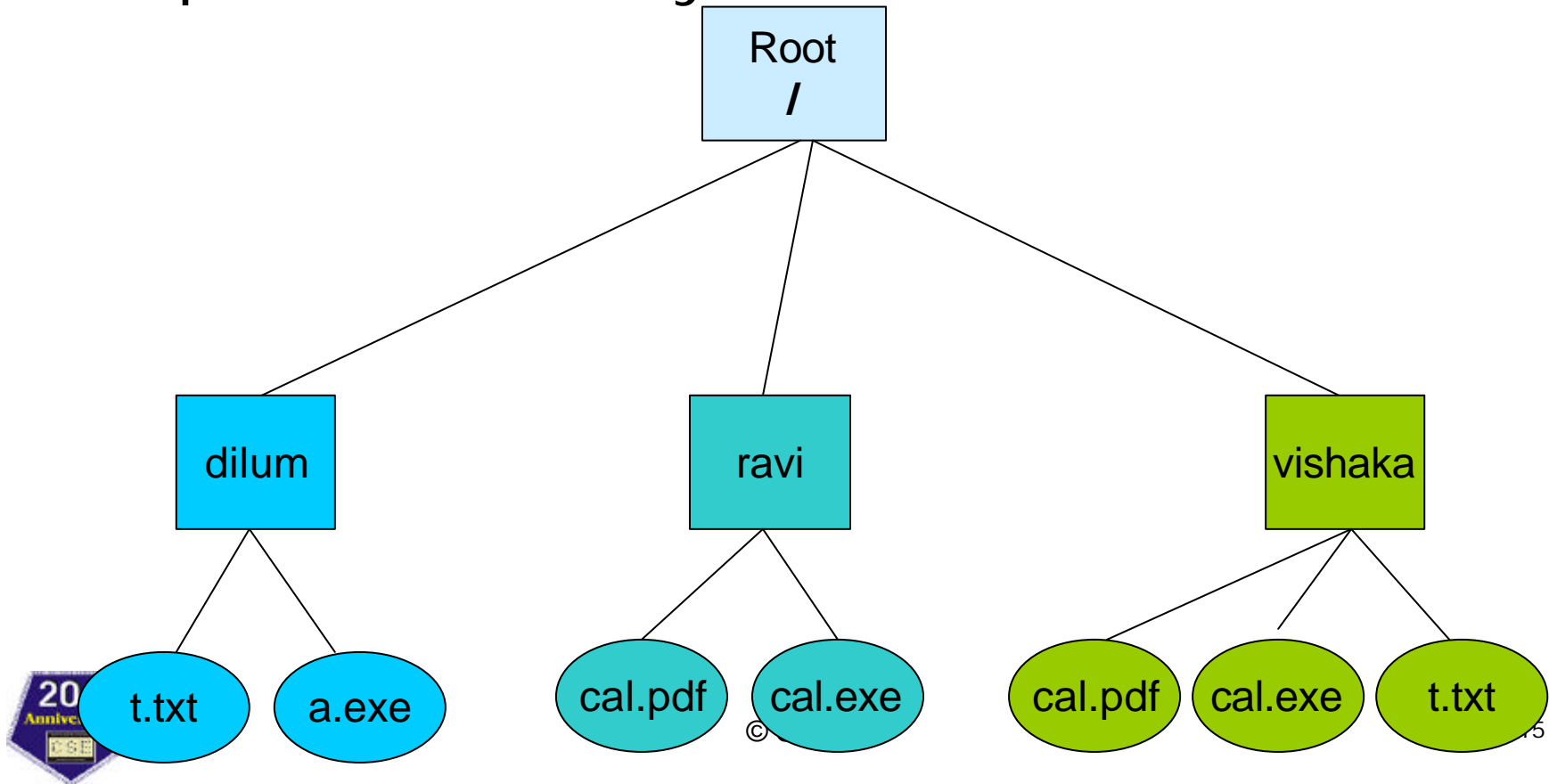
---

- ❑ Simplest form of directory system where 1 directory contains all the files
- ❑ This single directory is called the **root**
- ❑ Problems – in a multi-user systems users can't have files with same name



# Directories - Two Level Systems

- To avoid the conflict each user is given a separate directory



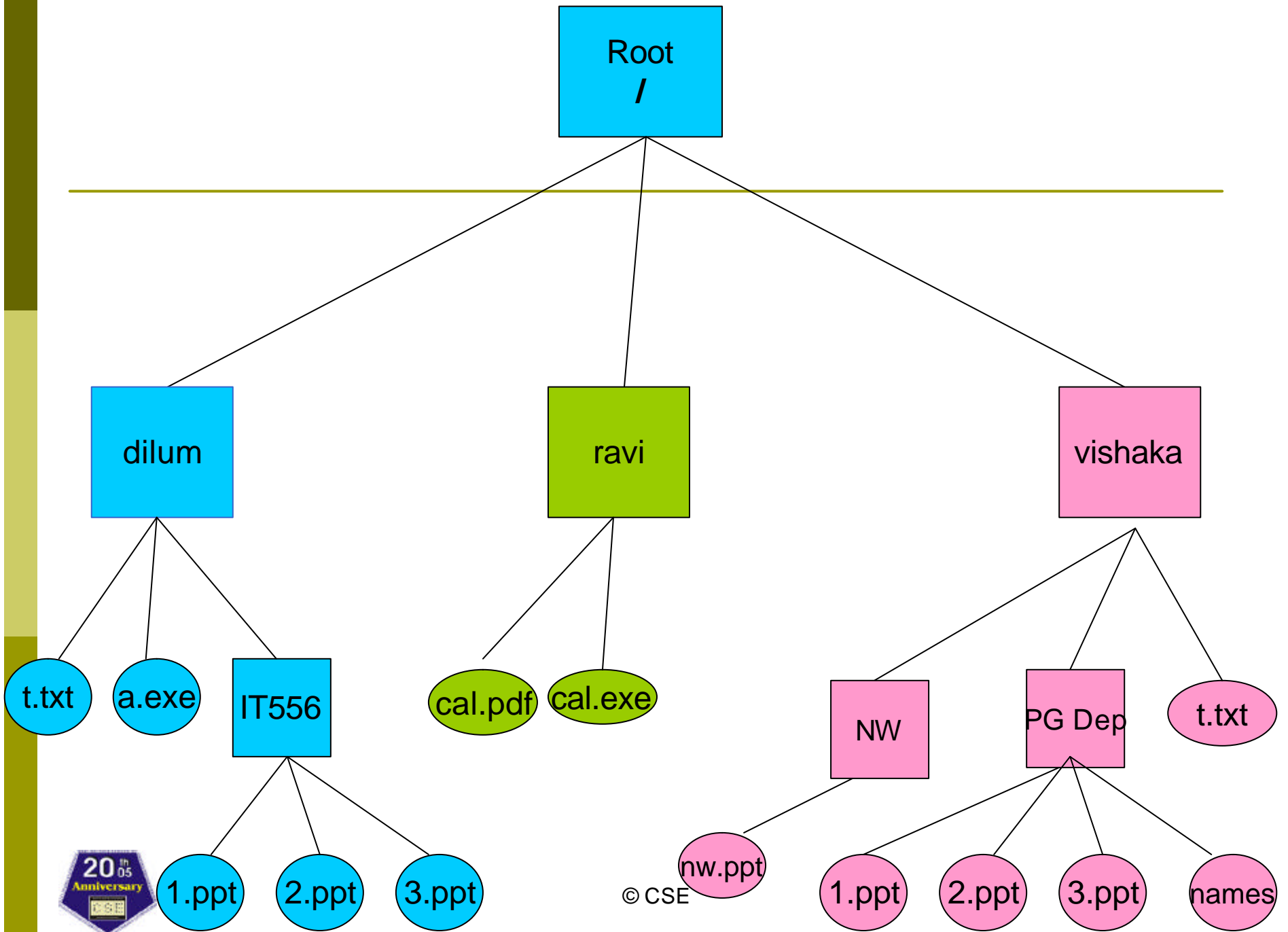
# Directories

## – Hierarchical Directory Structure

---

- ❑ Two Level directory structure is not enough when users want to manage their own files
- ❑ All most all the commercial OS supports multiple directory levels
- ❑ However CD-ROM file system has a limit in number of levels in the hierarchy





# Directory Operations

---

- ❑ **Create** – create a new directory
- ❑ **Delete** – delete an existing directory
- ❑ **Opendir** – open the directory for reading
- ❑ **Readdir** – read the contents of the directory
- ❑ **Closedir** – close the directory
- ❑ **Rename** – change the name of the directory
- ❑ **Link** – allow files to appear in more than one directory. Related to file sharing



# Path Names

---

- ❑ When files are in a directory tree there should be a mechanism to name them
- ❑ Absolute path names
  - Path from the root directory to the file
  - `/vishaka/NW/nw.ppt`
- ❑ Relative path names
  - Give relative to the current working directory
  - If currently in NW directory path name is `nw.ppt`
  - If currently in `vishaka` directory path name is `NW/nw.ppt`



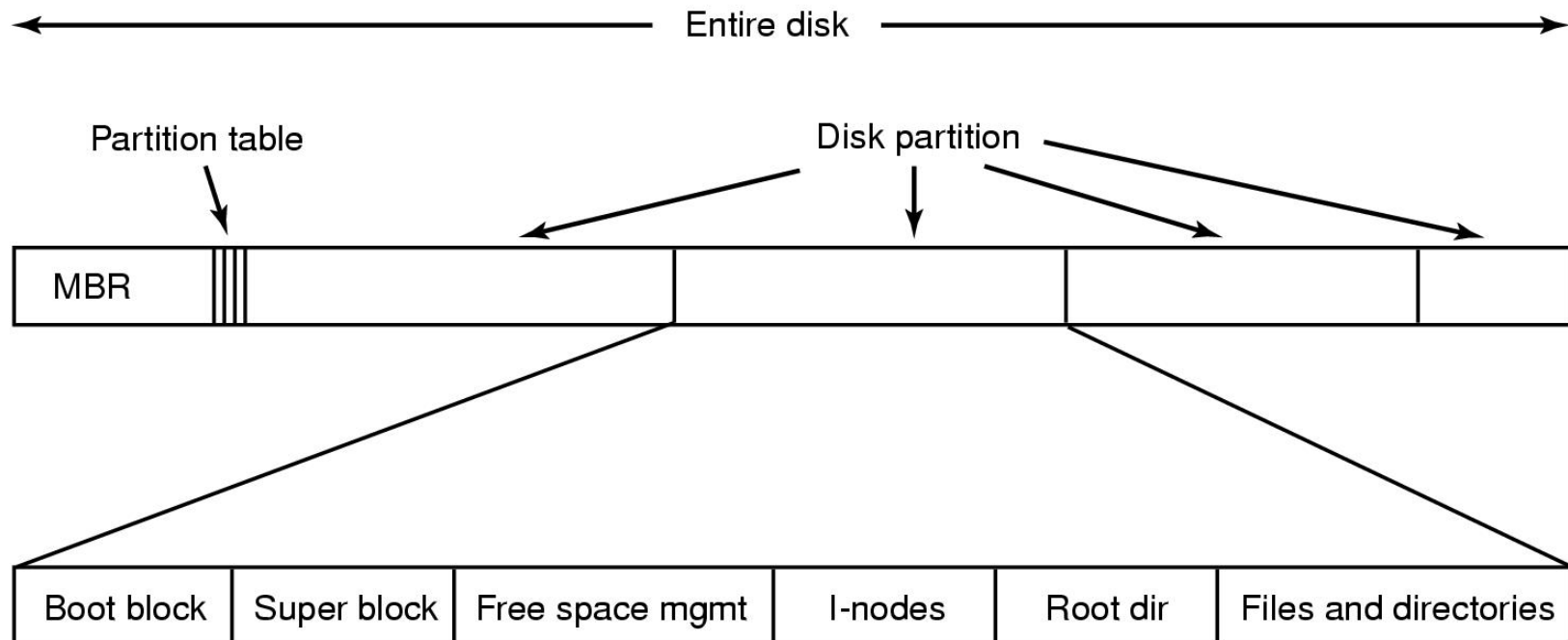
# Path Names cont...

---

- ❑ Regardless of the current working directory absolute pathnames will always work
- ❑ There are 2 special entries in each directory
  - . (dot) – refers to the current working directory
  - .. (double dot/dotdot)– refers to the parent directory
  - Example : `./NW/nw.ppt`



# File System layout



# Implementing Files

---

- ❑ Need to keep track of where a file is located on the disk
- ❑ Files are stored as blocks
- ❑ Several approaches are used to store & keep track of files
  1. Contiguous allocation
  2. Link list allocation
  3. Link list allocation using a table in memory
  4. I –nodes (will not study)



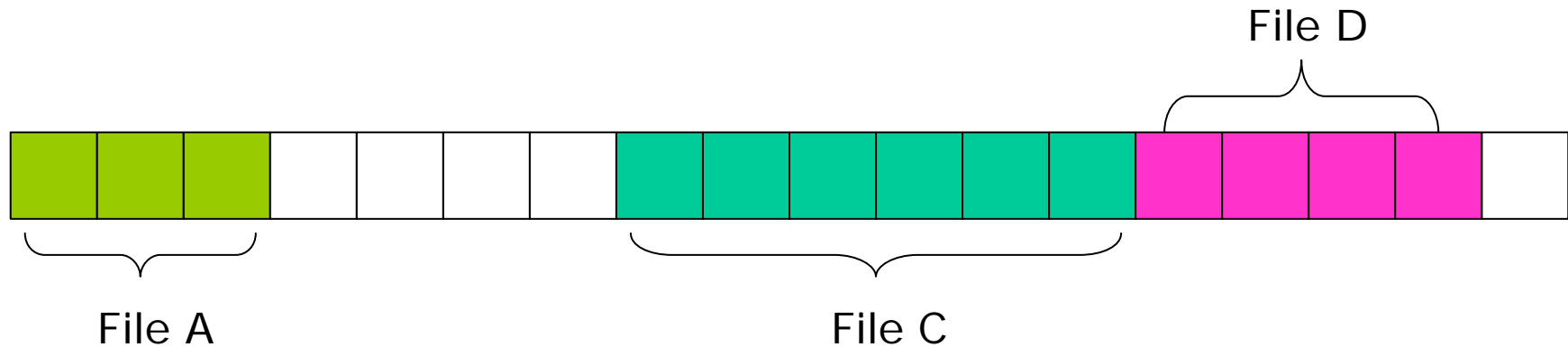
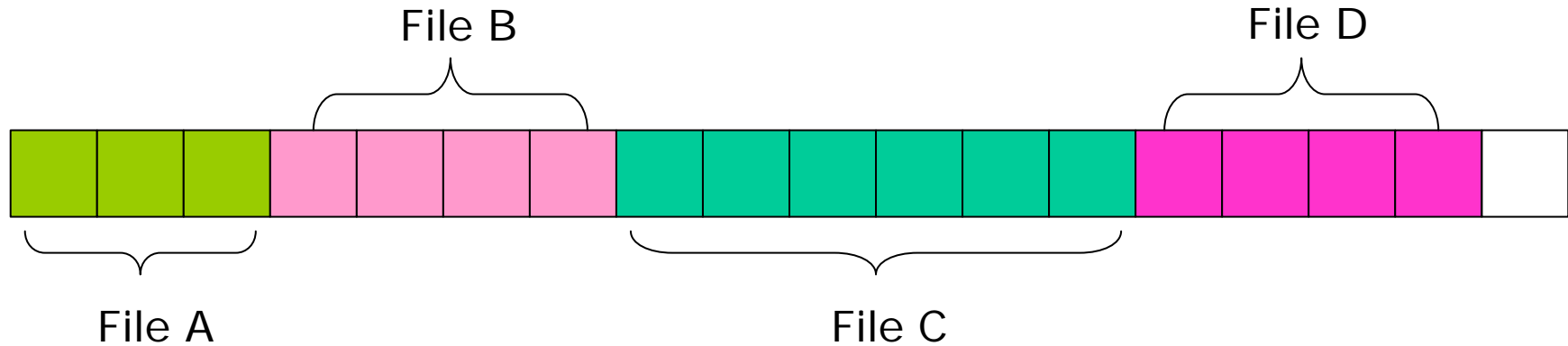
# Contiguous Allocation

---

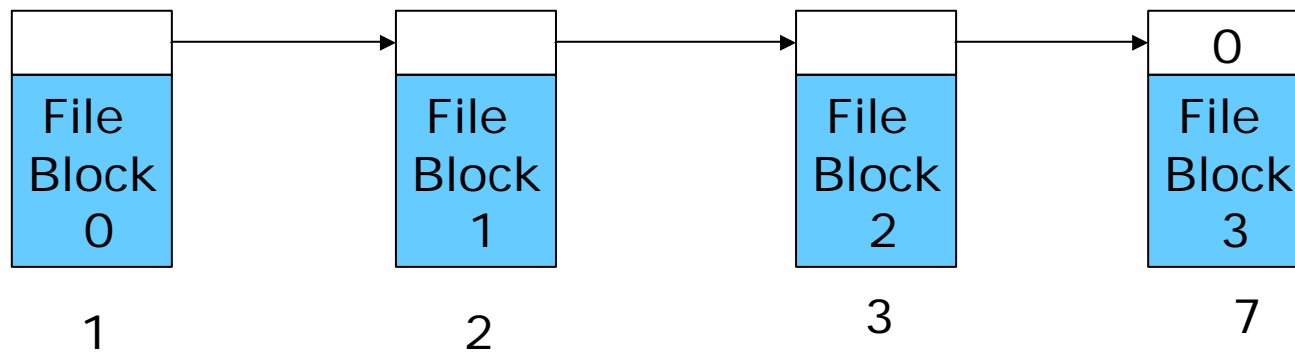
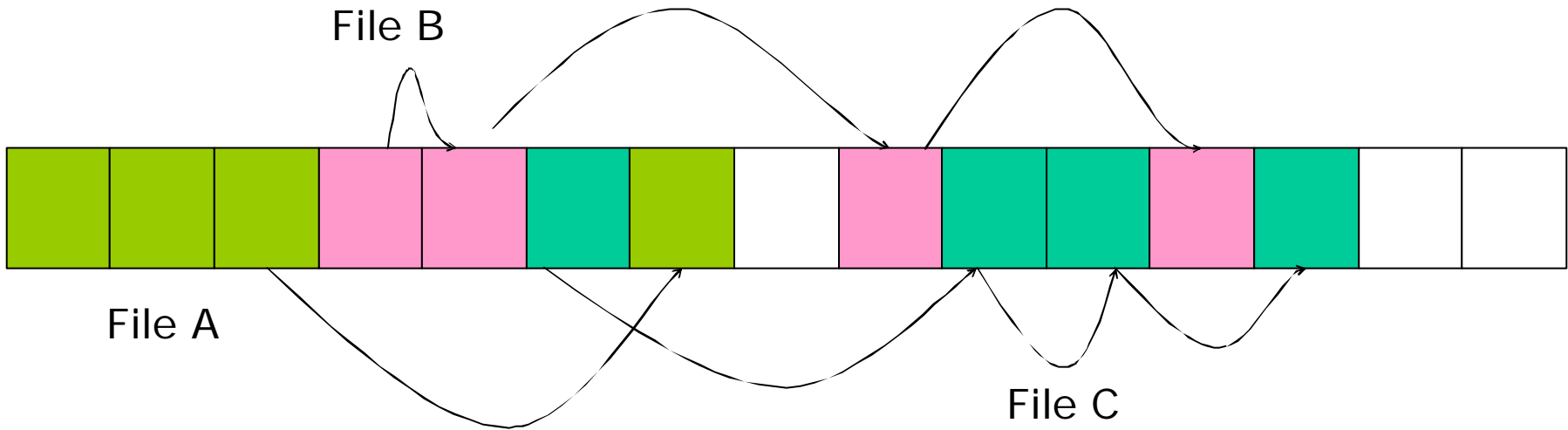
- ❑ Is the simplest allocation scheme
- ❑ One file is stored after another file
- ❑ Advantages
  - Simple to implement
  - Can be read much faster
- ❑ Disadvantages
  - Disk fragmentation
- ❑ Not used in commercial OSs but suitable for Embedded OSs & CD-ROMs



# Contiguous Allocation cont...



# Linked List Allocation



Physical  
block

1

2

3

7



# Linked List Allocation cont...

---

## □ Advantages

- Every disk block can be used
- No space is lost due to fragmentation

## □ Disadvantages

- Random access is slow
- If the link is lost rest of the file can't be located



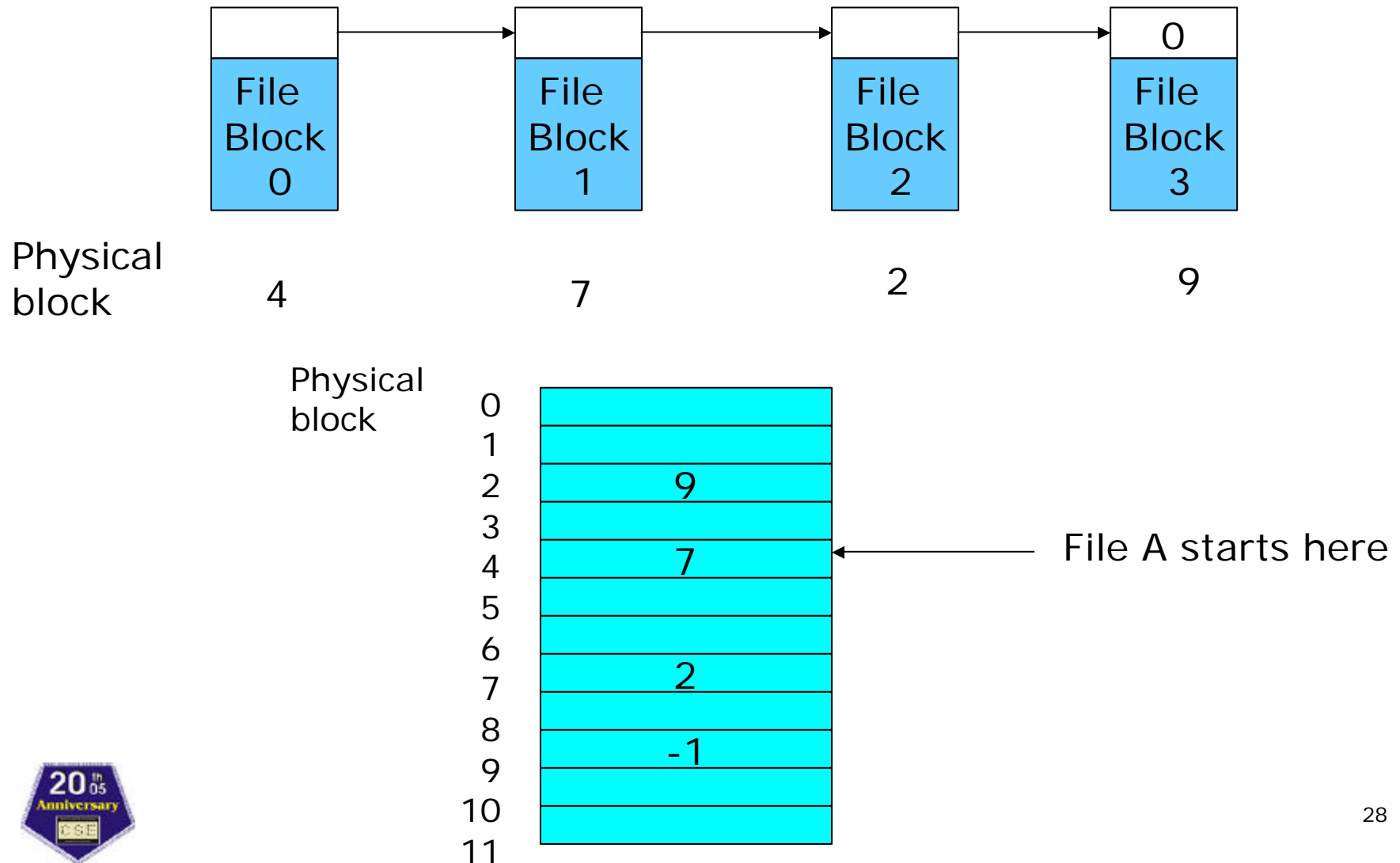
# File Allocation Tables - FAT

---

- ❑ Put the link list allocation entries in memory
- ❑ Advantages
  - Fast random access
  - If one of the blocks in the disk is lost still the rest of the file can be located
- ❑ Disadvantages
  - FAT takes some space in memory



# File Allocation Tables – FAT cont...



# DOS File System

---

- ❑ Make use of File Allocation Table
- ❑ Use of 8+3 character file names
- ❑ Attributes
  - Read only, archived, hidden, system
- ❑ 2 versions
  - FAT-12
  - FAT-16



# DOS File System cont...

---

- FAT-12
  - Max partition size 2MB
  - 4 separate partitions
- FAT-16
  - Max partition size 2GB
  - 4 separate partitions



# Windows 98 File System

---

- ❑ Actually came with Windows 95, 2<sup>nd</sup> release
- ❑ Namely FAT-32
  - Max partition size 2TB
  - More than 4 partitions
- ❑ File names up to 255 characters
- ❑ Was backward compatible with FAT-16



# Windows 2000 File System

---

- ❑ NTFS – New Technology File System
- ❑ Initially used in Windows NT
- ❑ A single partition can be up to  $2^{64}$  bytes
- ❑ File names can be up to 255 Unicode characters
- ❑ File system security is inbuilt
- ❑ Better performance & more reliable
- ❑ Not backward compatible with FAT-16 or FAT-32



# UNIX/Linux File Systems

---

- ❑ Make use of I-nodes
- ❑ Initially supported only 14 characters but later versions support 255 characters
- ❑ File system security is inbuilt
- ❑ Supports many file systems
  - V7
  - ext, ext2, ext3
  - NFS – Network File System
  - VFAT – UNIX version of FAT



# Ext File System

---

## □ ext

- Supports 14 characters
- A single file can be up to 2GB
- Later better versions were introduced such as ext2, ext3

# Summary

---

- ❑ What is a file & file system
- ❑ Files & Directories
- ❑ File naming, types, structure, access
- ❑ File & Directory attributes & operations
- ❑ Directory hierarchies
  - Single level, two level & hierarchical directory systems
- ❑ Path Names
- ❑ File systems
  - FAT-12, FAT-16, FAT-32, NTFS, ext

